Technical Bulletin

Part No. 00D-TB025

DataStage Teradata MultiLoad/TPump/FastExport

This technical bulletin describes and explains the use of Version 1.3 of DataStage Teradata MultiLoad/TPump/FastExport. The stage uses the Teradata MultiLoad utility or the Teradata TPump utility to load data into a Teradata database. It uses the Teradata FastExport utility to unload data from a Teradata database.

Printing History

First Edition (00D-TB025) for Version 1.2, September 2002
Updated for Version 1.2, March 2003
Updated for Version 1.2, August 2003
Second Edition for Version 1.3, December 2003

How to Order Technical Documents

To order copies of documents, contact your local Ascential subsidiary or distributor, or call our office at (508) 366-3888.

Documentation Team: Marie E. Hedin

Introduction

This technical bulletin describes the following for Version 1.3 of the DataStage Teradata MultiLoad/TPump/FastExport Plug-in for DataStage 7.0.1:

DataStage provides the ability to use Teradata's MultiLoad, TPump, and FastExport utilities by generating the scripts that import or export data to or from a Teradata database. Reference links have no meaning in the context of this stage and are not allowed.

Input Link. Utilizes the MultiLoad utility to perform fast, high-volume maintenance functions on multiple tables and views of a Teradata database. Utilizes the TPump utility to do concurrent updates on the same table. DataStage Teradata MultiLoad/TPump/FastExport generates a MultiLoad script or a TPump script allowing you to run the load utility through DataStage or invoke the load utility manually.

Output Link. Utilizes the FastExport utility to transfer large amounts of data quickly from tables and views of a Teradata database. DataStage Teradata MultiLoad/TPump/FastExport generates a FastExport script allowing you to run FastExport through DataStage.

Functionality

The DataStage Teradata MultiLoad/TPump/FastExport plug-in has the following functionality:

- Support for data files which exceed the 2-GB file size limit for 64-bit file systems.
- Support for MetaStage. For additional information, see *MetaStage User's Guide*.
- Generation, and optional automatic execution, of the Teradata commands:
  - To load or update a database with data from input links (MultiLoad or TPump).
  - To unload data from a database to output links (FastExport).

- Two load modes: manual or automatic.
- Load parameters to control the load process.
- NLS (National Language Support). For additional information, see *DataStage NLS Guide*.

The following functionality is not supported:

- Compatibility with DataStage releases before 7.0
- Reference output links
- Meta data importing
- Support for stored procedures
- Native data browsing
- Support for reject row handling

Installing the Plug-In

For instructions and information supporting the installation, see *DataStage Plug-In Installation and Configuration Guide*.

Before installing the Plug-in, consult Teradata documentation for any specific configuration requirements.

Building a MultiLoad or TPump Script

You need to specify the stage properties when you build a MultiLoad or TPump script. To do this:

1. Create a DataStage job.
2. Define the properties used by the Designer to construct the script.
3. Compile and run the job.

Each task is described in more detail in the following sections.

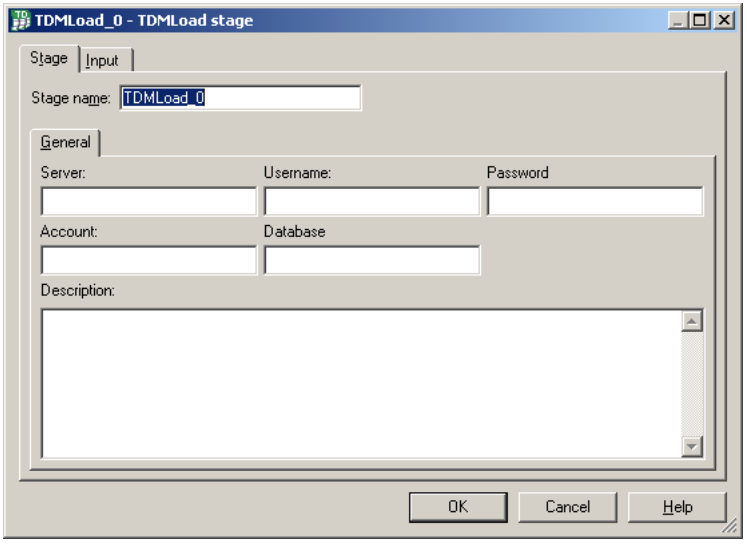
Creating a DataStage Job

1. Create a job using the DataStage Designer. For more information, see *DataStage Designer's Guide.*

2. Choose **TDMLoad** from the **Insert** menu or select the icon from the Designer job palette.
3. Add an input stage and add a link.

Defining Properties

Right click the **TDMLoad** icon and select **Properties** or choose **Properties** from the **Edit** menu. The **TDMLoad Stage** dialog box appears:



This dialog box has two pages:

- **Stage**. Displays the name of the stage you are editing. The **General** tab defines the Teradata data target and logon information.
- **Input**. Specifies the information necessary to generate a MultiLoad or TPump script. This page also specifies the associated column definitions.

About the Stage Page

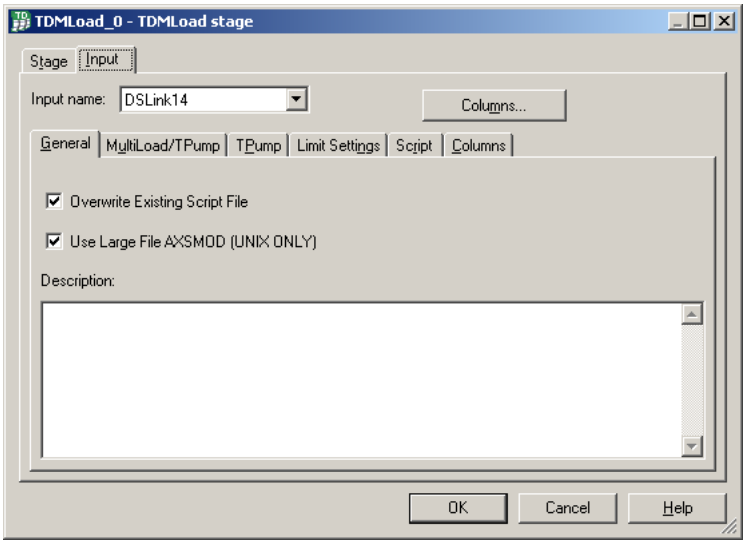
Supply information on the **General** tab on the **Stage** page to identify the target of the import. The **General** tab supports the following properties:

Server	The name of a Teradata Director Program (TDP). Optional.

Username	A name that identifies the user. The user must have the necessary privileges to write to the database. Required.
Password	The password associated with the Username. Required.
Account	The account associated with the Username. Optional.
Database	The name of the database to be loaded or updated. Optional.
Description	A description of the stage. Optional.

About The Input Page

The **Input** page has an **Input name** field, a **Columns...** button, and **General**, **MultiLoad/TPump**,**TPump**, **Limit Settings**, **Script**, and **Columns** tabs.

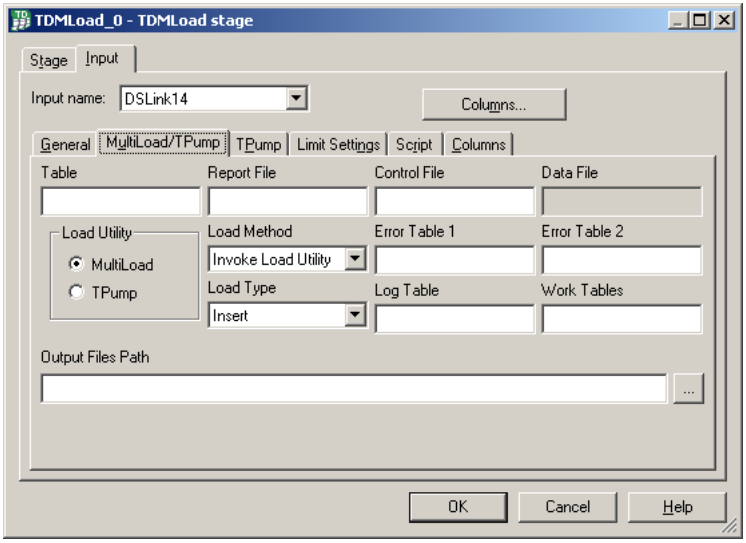


- **Input name**. The name of the input link. Choose the link you want to edit from the **Input name** list. This list displays all the input links to the TDMLoad stage.
- Click **Columns...** to display a brief list of the columns designated on the input link. As you enter detailed meta data on the **Columns** tab, you can leave this list displayed.

General Tab. The following properties are included on the **General** tab:

Overwrite Existing Script File	The current script is replaced each time the job is compiled. If you want to make and save modifications directly to the generated script, clear **Overwrite Existing Script File**. The default is **Overwrite Existing Script File** selected.
Use Large File AXSMOD (UNIX ONLY)	This property, available on UNIX platforms only, allows the processing of files larger than 2 gigabytes. This is an extension to MultiLoad and TPump. To use this feature, you must have *AXSMOD lf_AXSMOD.so* (or *.sl*) installed on the server. This file is provided by Teradata. The default is **Use Large File AXSMOD** selected.
Description	A description of the script. Optional.

MultiLoad/TPump Tab. Use the **MultiLoad/TPump** tab to provide general information about the script to be generated.



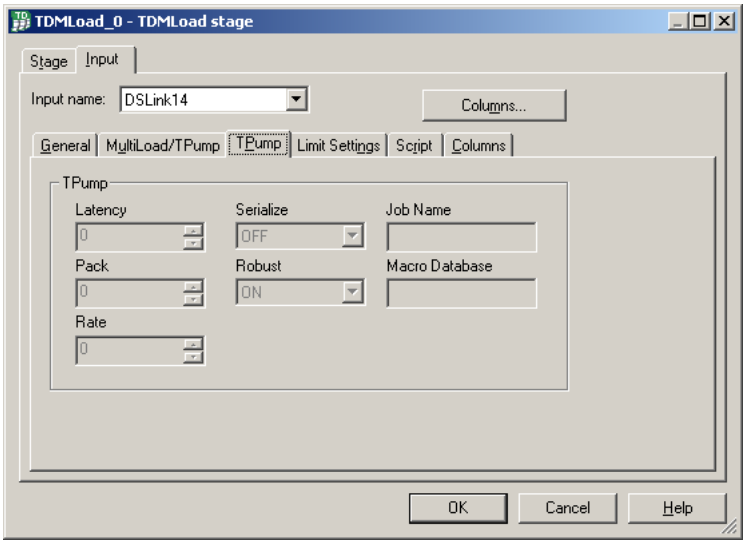
The following properties are included on the **MultiLoad/TPump** tab:

Table	The name of the table to be loaded. The name is used in the BEGIN MLOAD command in the generated script. Required.

Load Utility	Two mutually exclusive option buttons used to specify either the MultiLoad utility or the TPump utility as the load method. The default is **MultiLoad** selected.
Report File	The name of the report file created by the load utility at execution time. The default name is table.txt, where table is the name supplied in the **Table** text box, or database_table.txt, if you supply a database name in the **Database** text box.
Control File	The name of the generated script. The default name is table.fl, where table is the name supplied in the **Table** text box, or database_table.fl, if you supply a database name in the **Database** text box.
Data File	The name of the data file if you select the manual load. See Load below. The default name is table.dat, where table is the name supplied in the **Table** text box, or database_table.dat, if you supply a database name in the **Database** text box.
Load Method	The timing of the execution of the script. If you select **Invoke Load Utility**, the load utility is executed automatically when the job is run. The stage creates a named pipe to transmit data to the load utility, and then it starts the load process. The load process has 12 minutes (720 seconds) to reach the phase in which the utility opens the named pipe to read the data. This is called the Acquisition Phase. If the load utility fails to reach Acquisition Phase and open the named pipe before the time has expired, the stage aborts and stops the process. If the default timeout is insufficient, you can override the default by doing one of the following: • Define the following environment variable in the DataStage Administrator. **DS_TDM_PIPE_OPEN_TIMEOUT** *n* *n* is the number of seconds the stage waits before aborting. • Use the Tenacity option on the **Limit Settings** tab. If you select **Manual**, the data to be imported is stored as a .dat file. You can then execute the script, which points to the .dat file, and load the data independent of the DataStage job. The default is **Invoke Load Utility** selected. See "Data File" above.

Load Type	The type of load activity taking place when the load utility selected in **Load Utility** executes. The options are: • Insert - Instructs the load utility to insert all acceptable rows. • Update - Instructs the load utility to update any rows for which there is a key match. • Delete - Instructs the load utility to delete any rows for which there is a key match. • Upsert - Instructs the load utility to update any rows for which there is a key match and insert any acceptable rows for which there is no key match. • Custom - Generates a custom script to load data. See "Script Tab" on page 10. The default is **Insert** selected.
Error Table 1	The name of an error table used by the load utility when it detects errors during execution of the script. See your Teradata MultiLoad or TPump documentation.
Error Table 2	The name of a second error table used by MultiLoad when it detects errors during execution of the script. TPump does not use a second error table, and this text box is unavailable if you select the **TPump** option button. See your Teradata MultiLoad documentation.
Log Table	The name of the log table used by the load utility if the script includes a .log table command.
Work Tables	The names of the special unhashed tables used by the MultiLoad utility when executing both import and delete tasks. TPump does not use work tables, and this text box is unavailable if you select the **TPump** option button. See your Teradata MultiLoad documentation.
Output Files Path	The name of the path to be used for the Report File, the Control File, and the Data File. Use the **Browse...** button to facilitate identifying the path.

TPump Tab. This tab contains options that apply only to the TPump utility. These options are not available if you select the **MultiLoad** option button.

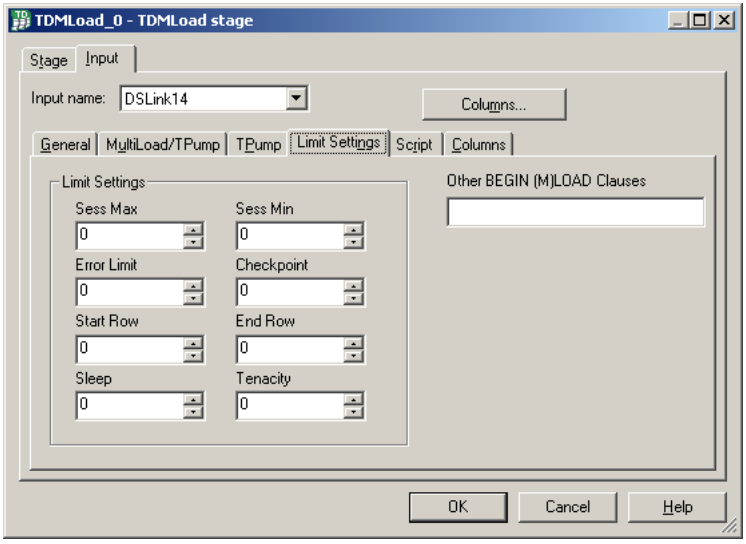


The following properties are included on the **TPump** tab:

Latency	The maximum number of seconds that a record resides in the TPump buffers before the buffers are flushed. The default value of 0 indicates the Teradata default should be used.
Pack	The number of statements to pack into a multiple-statement request. The default value of 0 indicates the Teradata default should be used.
Rate	The number of statements per minute to be sent to the Teradata database. The default value of 0 indicates the Teradata default should be used.
Serialize	The way multiple operations on a given row are guaranteed to occur. If **On** is selected, the multiple operations occur serially. The default is **Off** selected.
Robust	The restart logic to be used. If **Off** is selected, TPump uses simpler but less reliable restart logic. The default is **On** selected.

Job Name	A unique identifier assigned to the TPump environment variable SYSJOBNAME. TPump truncates the identifier to 16 characters. If not provided, TPump uses the Teradata default.
Macro Database	The name of the database that is to contain any macros built or used by TPump. If not provided, TPump uses the Teradata default.

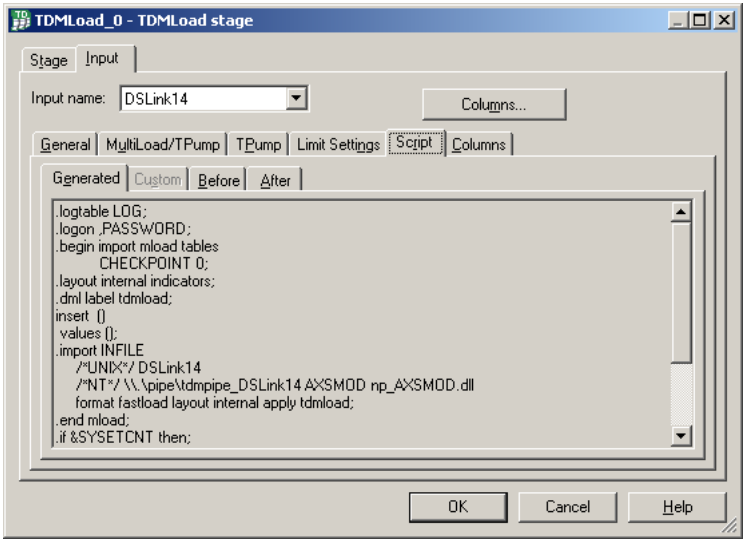
Limit Settings Tab. Limit Settings are used in the BEGIN MLOAD or BEGIN LOAD command and correspond directly to options in the command.



The following properties are included on the Limit Settings tab:

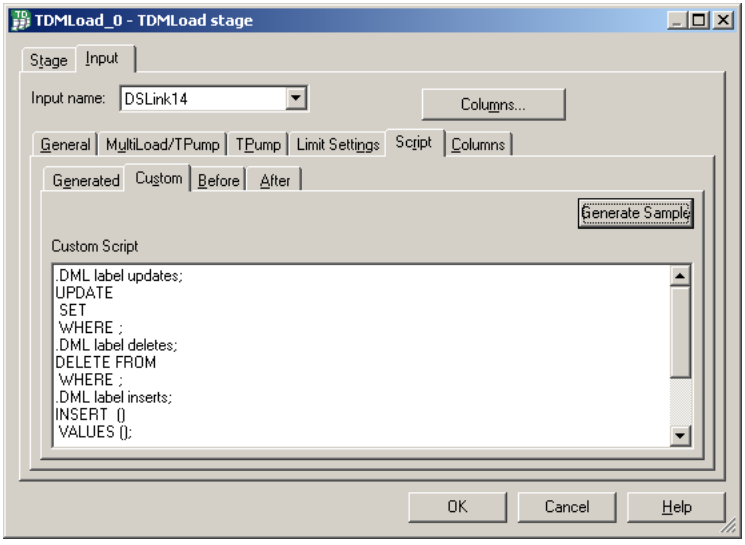
Limit Settings	The settings include: • Sess Max • Sess Min • Error Limit • Checkpoint • Start Row • End Row • Sleep • Tenacity See your Teradata MultiLoad or TPump documentation for descriptions of these settings. If you select **TPump** on the **MultiLoad/TPump** tab (see "MultiLoad/TPump Tab" on page 5), you must specify a value in **Sess Max** greater than 0.
Other BEGIN (M)Load Clauses	You can specify additional clauses for the .BEGIN LOAD or .BEGIN MLOAD statement that are not covered by the other properties.

Script Tab. Use the **Script** tab to build your own script to run the load utility. This capability allows you to take advantage of the power of MultiLoad or TPump and create a script having some combination of the Update, Insert, Delete, and Upsert load types.



The **Script** tab consists of four tabs: **Generated**, **Custom**, **Before**, and **After**.

- **Generated tab**. The **Generated** tab displays the generated MultiLoad or TPump script.
- **Custom tab**. Build your custom script in the **Custom Script** text box.



Start with the DML LABEL command and end with the IMPORT command. You can customize statements between the .BEGIN LOAD and .END LOAD statements (TPump) or between the .BEGIN MLOAD and .END MLOAD statements (MultiLoad). You must also supply column names and values for each option you choose. See your Teradata MultiLoad or TPump documentation for additional information.

The stage provides assistance in creating a custom script. Click **Generate Sample** to populate the text box with a sample script based on the meta data supplied to the stage. The generated sample will *not* run unless it is modified. At a minimum you must activate the section beginning .import:

– If you are working in a UNIX environment, change the information immediately to the right of /*UNIX*/ by removing the "/*" and "*/."

For example, change the line

```
/*UNIX*/ /*DSLINK11*/
```

to

```
/*UNIX*/ DSLINK11
```

– If you are working in a Windows NT environment, change the information immediately to the right of /*NT*/ by removing the "/*" and "*/." For example, change the line

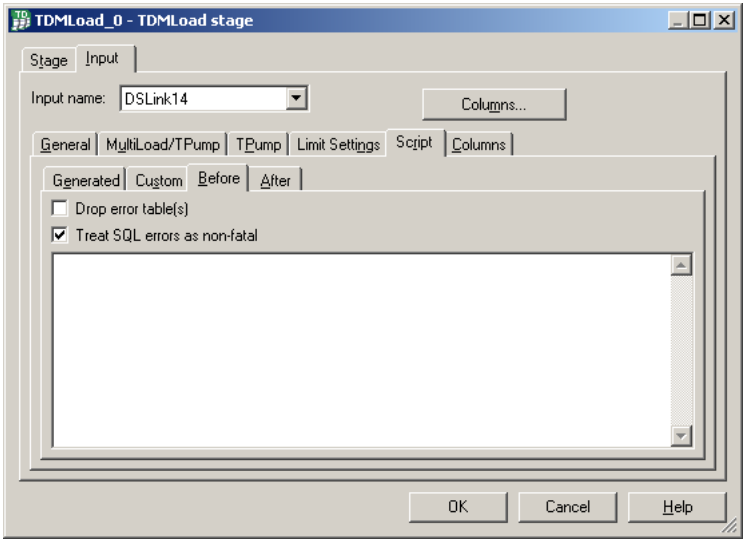
```
/*NT*/ /*\\.\pipe\tdmpipe_DSLINK11 AXSMOD np_AXSMOD.dll*/
```

to

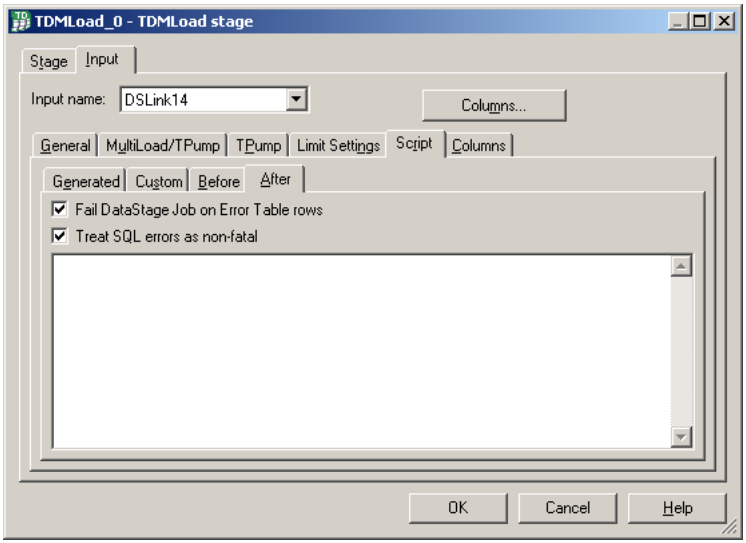
```
/*NT*/ \\.\pipe\tdmpipe_DSLINK11 AXSMOD np_AXSMOD.dll
```

- **Before and After tabs**. The **Before** and **After** tabs allow additional customization. They apply to both MultiLoad and TPump and to every Load Type: Insert, Update, Delete, Upsert, and Custom.

  – **Before tab**. Insert custom statements before the .BEGIN LOAD or .BEGIN MLOAD statement in the generated script using the text box.



– **After tab**. Insert custom statements after the .END LOAD or .END MLOAD statement in the generated script using the text box.



A statement can span multiple lines. Each statement must begin on a new line and end with a semicolon. MultiLoad or TPump commands that are not SQL statements must begin with a period.

If **Drop error table(s)** is selected, the generated script drops the error tables before loading begins. If **Drop error table(s)** is not selected, which is the default, the Plug-in assumes one of the following:

– There are no error tables to drop. Therefore, the Plug-in does not put statements to drop the error tables in the generated script. If either the log table or error tables preexist, MultiLoad or TPump fails.

– You are attempting a restart. Therefore the log table and error tables must preexist from a previous run.

In order to drop the error table, do the following:

– If you select **TPump** on the **MultiLoad/TPump** tab, provide a name in either **Error Table 1** on the **MultiLoad/TPump** tab (see "MultiLoad/TPump Tab" on page 5) or in **Job Name** on the **TPump** tab (see "TPump Tab" on page 8). If you do not provide a name in **Error Table 1**, TPump uses *JobName*_ET as the error table name.

– If you select **MultiLoad** on the **MultiLoad/TPump** tab, provide names in **Error Table 1** and **Error Table 2** (see "MultiLoad/TPump Tab"on page 5). If you do not provide names in **Error Table 1** and **Error Table 2**, Multi-Load uses ET_*TableName* and UV_*TableName* as the error table names (see "MultiLoad/TPump Tab" on page 5).

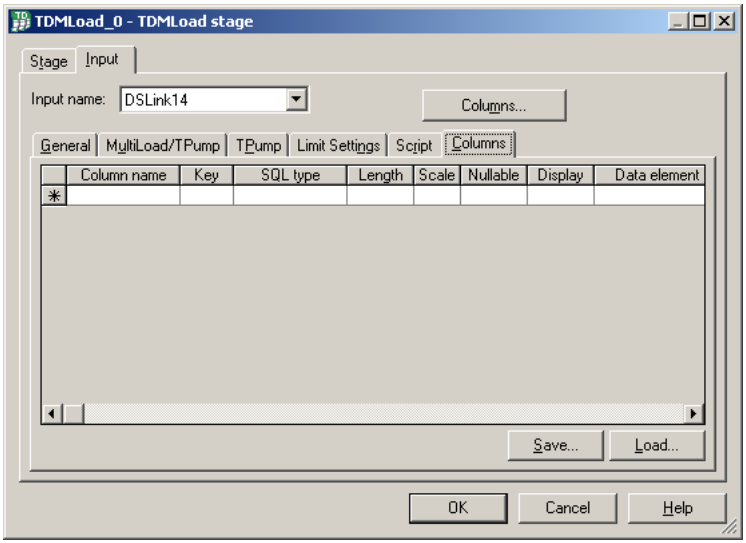
The following property is included on the **Before** and **After** tabs:

Treat SQL errors as non-fatal	The conditions under which processing stops with regard to Before and After SQL statements. If **Treat SQL errors as non-fatal** is selected, errors caused by Before SQL statements or After SQL statements are ignored, and processing continues. If **Treat SQL errors as non-fatal** is not selected, errors from SQL statements are treated as fatal to the job. The default is **Treat SQL errors as non-fatal** selected.

The following property is include on the **After** tab:

Fail DataStage Job on Error Table rows	During the load process, MultiLoad populates two error tables if rows contain errors. Similarly, TPump populates one error table if rows contain errors. See "MultiLoad/TPump Tab" on page 5. If you select **Fail DataStage Job on Error Table rows**, the job aborts if the load utility inserts any rows into those tables. The default is **Fail DataStage Job on Error Table rows** selected.

Columns Tab. This tab contains the meta data for the data to be imported to the Teradata database.



You can manually enter the meta data in the grid or load it from the DataStage Repository. The columns listed determine what is imported to the Teradata database.

Click **Save** to save the meta data to the DataStage Repository. Click **Load** to load the meta data from the Repository.

Compiling and Running the Job

Complete the definition of the other stages in your job design according to normal DataStage procedures. Compile and run the job.

Building a FastExport Script

You need to specify the stage properties when you build an FastExport script. To do this:

1. Create a DataStage job.
2. Define the properties used by the Designer to construct the script.
3. Compile and run the job.

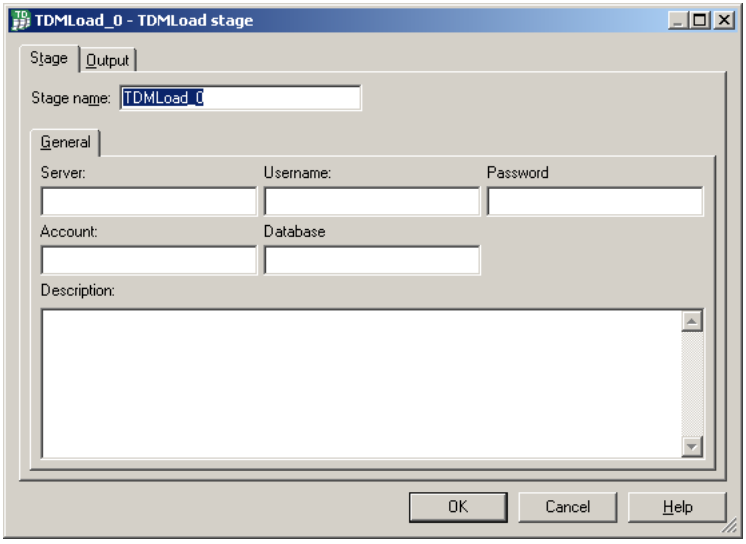
Each task is described in more detail in the following sections.

Creating a DataStage Job

1. Create a job using the DataStage Designer. For more information, see *DataStage Designer's Guide.*
2. Choose **TDMLoad** from the **Insert** menu or select the icon from the Designer job palette.
3. Add an output stage and add a link.

Defining Properties

Right click the **TDMLoad** icon and select **Properties** or choose **Properties** from the **Edit** menu. The **TDMLoad Stage** dialog box appears.



This dialog box has two pages:

- **Stage**. Displays the name of the stage you are editing. The **General** tab defines the Teradata data source and logon information
- **Output**. Specifies the information necessary to generate a FastExport script. This page also specifies the associated column definitions.

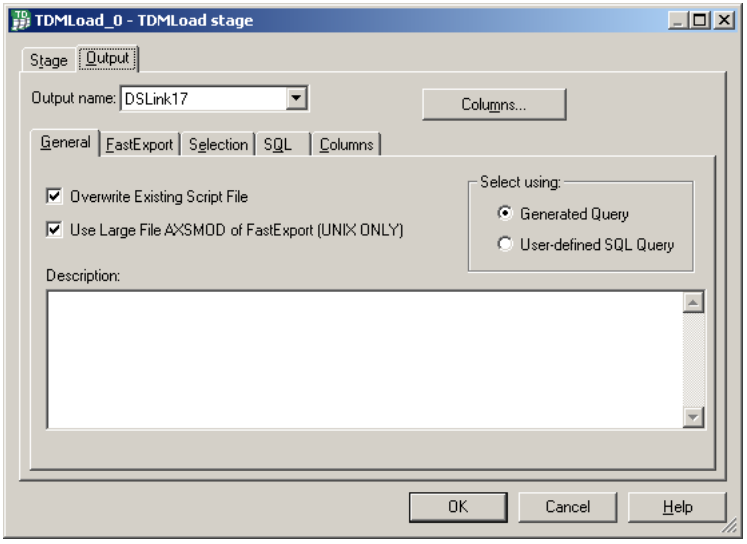
About the Stage Page

Supply information on the **General** tab on the **Stage** page to identify the target of the import. The **General** tab supports the following properties:

Server	The name of a Teradata Director Program (TDP). Optional.
Username	A name that identifies the user. The user must have the necessary privileges to read from the database. Required.
Password	The password associated with the Username. Required.
Account	The account associated with the Username. Optional.
Database	The name of the database from which data is to be exported. Optional.
Description	A description of the stage. Optional.

About the Output Page

The **Output** page has an **Output name** field, a **Columns...** button, and **General**, **FastExport**, **Selection**, **SQL**, and **Columns** tabs.

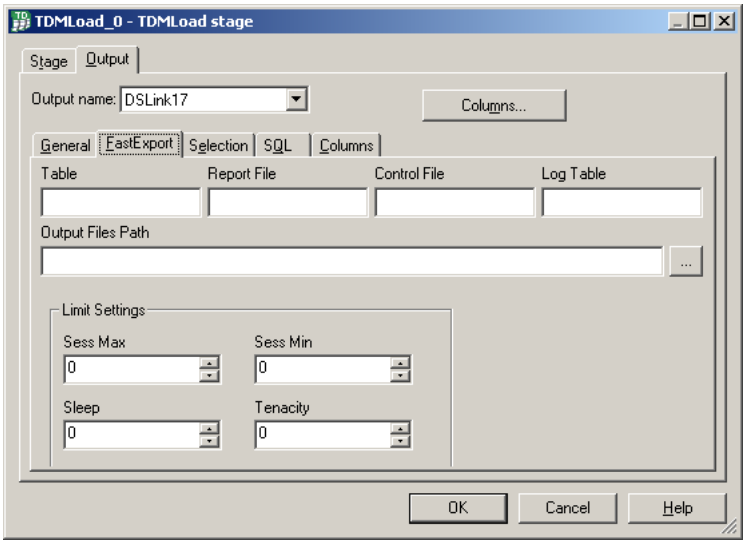


- **Output name**. The name of the output link. Choose the link you want to edit from the **Output name** list. This list displays all the output links from the TDMLoad stage.
- Click **Columns…** to display a brief list of the columns designated on the output link. As you enter detailed meta data on the **Columns** tab, you can leave this list displayed.

General Tab. The following properties are included on the **General** tab:

Overwrite Existing Script File	The current script is replaced each time the job is compiled. If you want to make and save modifications directly to the generated script, clear **Overwrite Existing Script File**. The default is **Overwrite Existing Script File** selected.
Use Large File AXSMOD of FastExport (UNIX ONLY)	This property, available on UNIX platforms only, allows the processing of files larger than 2 gigabytes. This is an extension to FastExport. To use this FastExport feature, you must have *AXSMOD lf_AXSMOD.so* (or *.sl*) installed on the server. This file is provided by Teradata. The default is **Use Large File AXSMOD of FastExport** selected. **Note:** If you are using Teradata client software TTU 7.0, do *not* select **Use Large File AXSMOD of FastExport**. Large File Access Module is obsolete with this release of Teradata client software. Failure to clear **Use Large File AXSMOD of FastExport** results in an error.
Select using Generated Query	The FastExport script is built using the SQL query generated from selection criteria. See "Selection Tab" on page 20. The default is **Generated Query** selected.
Select using User-defined SQL Query	The FastExport script is built using a manually created SQL query. See "SQL Tab" on page 21. The default is **User-defined SQL Query** cleared.
Description	A description of the script. Optional.

FastExport Tab. Use the **FastExport** tab to provide general information about the script to be generated.



The following properties are included on the **FastExport** tab:

Table	The name of the table to be exported. The name is used in the EXPORT command in the generated script. Required.
Report File	The name of the report file created by FastExport at execution time. The default name is table.txt, where table is the name supplied in the **Table** text box.
Control File	The name of the generated script. The default name is table.fl, where table is the name supplied in the **Table** text box.
Log Table	The name of the log table used by FastExport if the script includes a .log table command.
Output Files Path	The name of the path to be used for the Report File and the Control File. Use the **Browse…** button to facilitate identifying the path.

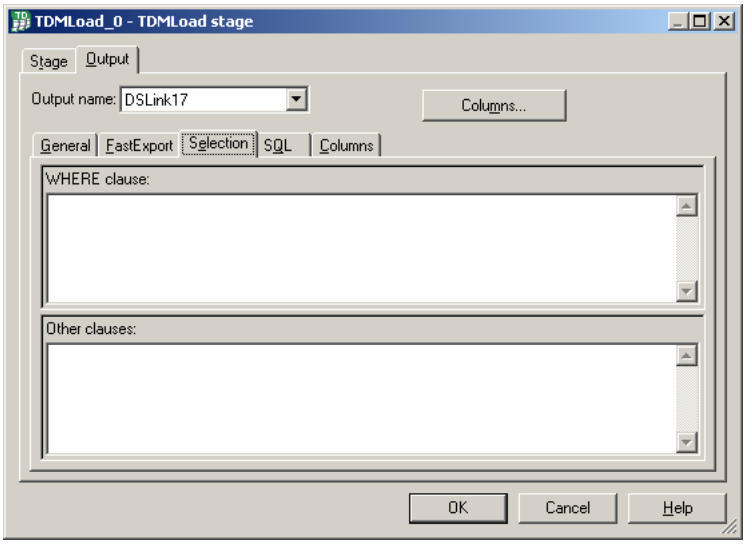
Limit Settings

Limit Settings are used in the BEGIN EXPORT command and correspond directly to options in the command. The settings include:

- Sess Max
- Sess Min
- Sleep
- Tenacity

See your Teradata FastExport documentation.

Selection Tab. Use this tab to specify the selection criteria used to generate the SQL query for the FastExport script.



The following properties are included on the **Selection** tab:

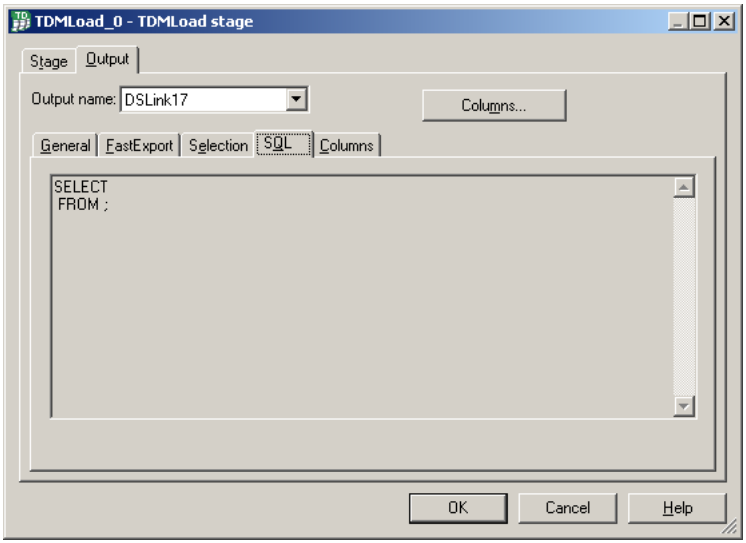
WHERE clause

An SQL WHERE clause specifying criteria the data must meet before being selected.

Other clauses

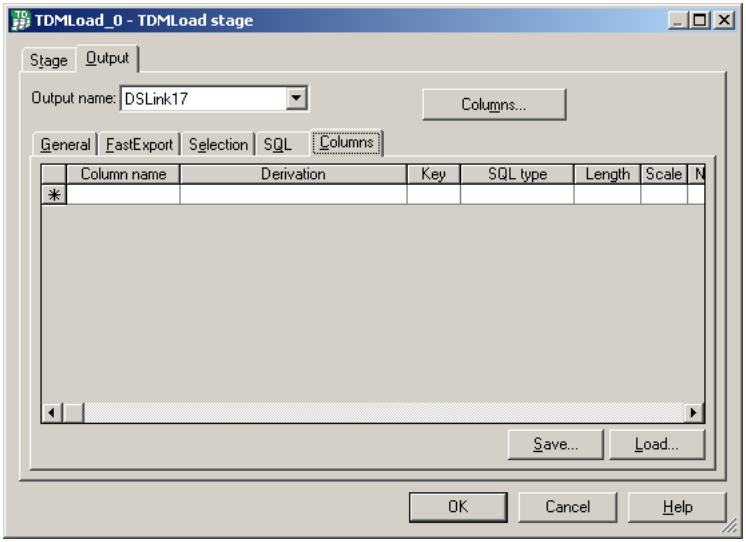
An optional GROUP BY, HAVING, or ORDER BY clause to sort, summarize, and aggregate data.

SQL Tab. Use this tab to construct the complete SQL Query for the generated FastExport script.



The SQL tab is active only if you choose **User-defined SQL Query** (see "Select using User-defined SQL Query" on page 18).

Columns Tab. This tab contains the meta data for the data to be exported from the Teradata database.



You can manually enter the meta data in the grid or load it from the DataStage Repository. The columns listed determine what is exported from the Teradata database. The **Column name** must be the same as the physical name or alias in the Teradata database or you must specify a **Derivation** that is the same as the physical name or alias in the Teradata database.

Click **Save** to save the meta data to the DataStage Repository. Click **Load** to load the meta data from the Repository.

Compiling and Running the Job

Complete the definition of the other stages in your job design according to normal DataStage procedures. Compile and run the job.

Writing Status Messages when Tracing Is Enabled

To aid with troubleshooting, you can write status messages to the DataStage log if tracing is enabled. See *Ascential DataStage Director Guide* for additional information about tracing.

- **To enable tracing when executing a job immediately.** In the DataStage Director, on the **Tracing** tab in the **Job Run Options** dialog box, select the active stages that connect to Teradata MultiLoad/TPump/FastExport. Then select **Subroutine calls**.
- **To enable tracing in all jobs.** In the DataStage Administrator, define the following environment variable and give it a value of 1.

  DS_TDM_TRACE_SUBROUTINE_CALLS 1

  To return to the default of tracing only when set from the **Job Run Options** dialog box, remove the variable or set it to 0.

In all cases, when tracing is turned off, the stage writes the following status messages to the DataStage log where UtilityName is either MultiLoad, TPump, or FastExport:

```
JobName..StageName: UtilityName process ProcessNumber has started

JobName..StageName: UtilityName has completed.  Report file: ReportFileName
```

When tracing is turned on, the stage writes the following MultiLoad or TPump status messages to the DataStage Log where LoadUtilityName is either MultiLoad or TPump:

```
JobName..StageName: Initializing link InputLinkName

JobName..StageName: Getting properties for link InputLinkName

JobName..StageName: Validating columns for link InputLinkName

JobName..StageName: Creating files for link InputLinkName

JobName..StageName: Starting process mload < ControlFileName >
ReportFileName 2>&1
```

or

```
JobName..StageName: Starting process tpump < ControlFileName >
ReportFileName 2>&1

JobName..StageName: LoadUtilityName process ProcessNumber has started

JobName..StageName: Opening file DataFileName

JobName..StageName: Started with the following parameters:  Server:
ServerName Username: UserName Password: Sorry its Encrypted Account:
AccountName Database: DatabaseName Control File: ControlFileName Table:
TableName Error Table 1: ErrorTable1Name

JobName..StageName: Continue with parameters: Error Table 2:
ErrorTable2Name Error Limit: 0 Sessions: 0 Row Start: 0 Row End: 0 Run
LoadUtilityName: Invoke LoadUtilityName Drop Table:  Dir path:  Base name:

JobName..StageName: Closing link InputLinkName
```

```
JobName..StageName: Closing file DataFileName

JobName..StageName: Waiting for process ProcessNumber to complete

JobName..StageName: Getting return code for process LoadUtilityName

JobName..StageName: Deleting file DataFileName

JobName..StageName: LoadUtilityName has completed.  Report file:
ReportFileName

JobName..StageName: Freeing resources for link InputLinkName
```

When tracing is turned on, the stage writes the following FastExport status messages to the DataStage log:

```
JobName..StageName: Initializing link OutputLinkName

JobName..StageName: Getting properties for link OutputLinkName

JobName..StageName: Validating columns for link OutputLinkName

JobName..StageName: Creating files for link OutputLinkName

JobName..StageName: Starting process fexp -r ".run file ControlFileName;" >
/dev/null 2>&1

JobName..StageName: FastExport process ProcessNumber has started

JobName..StageName: Opening file DataFileName

JobName..StageName: Closing link OutputLinkName

JobName..StageName: Waiting for process ProcessNumber to complete

JobName..StageName: Getting return code for process fexp

JobName..StageName: FastExport has completed.  Report file: ReportFileName

JobName..StageName: Closing file DataFileName

JobName..StageName: Deleting file DataFileName

JobName..StageName: Freeing resources for link OutputLinkName
```

Data Type Support

The following table documents the support for Teradata data types:

Data Type	Support
Unknown	Unsupported
BigInt	Unsupported
Binary	Unsupported

Data Type (Cont.)	Support (Cont.)
Bit	Supported
Char	Supported
Date	Supported
Decimal	Supported
Double	Supported
Float	Supported
Integer	Supported
LongNVarChar	Unsupported
LongVarBinary	Unsupported
LongVarChar	Unsupported
NChar	Unsupported
Numeric	Supported
NVarChar	Unsupported
Real	Supported
SmallInt	Supported
Time	Supported
Timestamp	Supported
TinyInt	Supported
VarBinary	Unsupported
VarChar	Supported

Overview of Stages Available for Teradata

DataStage supports several stages that access Teradata tables. The following table briefly describes each stage.

Stage	Functionality
DataStage Teradata Data Access	Provides access to Teradata for row-by-row operations using CLI (Call Interface). It is not efficient for bulk operations but works well for small table access.

Stage (Cont.)	Functionality (Cont.)
DataStage Teradata Bulk Load	Loads an empty Teradata table in bulk using the FastLoad command-line utility. This utility is faster than MultiLoad for this type of operation.
DataStage Teradata MultiLoad/TPump/ FastExport	Uses three different Teradata utilities: • MultiLoad - Inserts, updates, deletes, or upserts rows in a Teradata table using the MLOAD command-line utility. It can handle multiple SQL statements in a single operation. You can provide user-defined SQL for complex operations. This stage is best used for bulk updates, deletes, upserts, and complex interface operations. • TPump - Inserts, updates, deletes, or upserts rows in a Teradata table using the TPUMP command-line utility. It can do concurrent updates on the same table. It can handle multiple SQL statements in a single operation. You can provide user-defined SQL for complex operations. • FastExport - Exports data from a Teradata table using the FEXP command-line utility. You can customize it to use user-defined SQL statements for extraction. This stage is best used for bulk extracts from Teradata.